A METHODOLOGY FOR SERVICE ARCHITECTURES

Object Oriented Design works because an Object represents a real-world "thing".

Service Oriented Architecture works because a Service represents a real world "what we do".

Origin/Author	**:**	**STEVE JONES**
Additional	**:**	**Mike Morris**
Approved by	**:**	**Pam Maynard** **Carl Bate** **Andy Mulholland**
Approved for release		**26th October 2005**
Status	:	OASIS Draft

Capgemini UK plc
Floors 1-5
76 Wardour Street
London
W1F 0UU
Phone +44 (0)20 7734 5700
Fax +44 (0)20 7297 3900

Disclaimer

All advice given and statements and recommendations made in this document are:

(i) provided in good faith on the basis of information provided by third parties and/or otherwise generally available or known to Capgemini UK plc at the time of writing; and

(ii) made strictly on the basis that in no circumstances shall this document constitute or be deemed to constitute a warranty by Capgemini UK plc as to the accuracy or completeness of the contents of this document.. Capgemini UK plc shall not be liable for any loss, expense, damage or claim arising out of, or in connection with the contents of this document nor for any omission from it."

A METHODOLOGY FOR SERVICE ARCHITECTURES

Contents

THIS DOCUMENT CONTAINS 32 PAGES INCLUDING TITLE PAGE

A METHODOLOGY FOR SERVICE ARCHITECTURES

A METHODOLOGY FOR SERVICE ARCHITECTURES

1. Context

The purpose of this paper is to create "service discovery" approach that leads to the business service architecture of an enterprise or project. Its aim is to provide both the notation and approach for the creation of that architecture and to provide a notation that is not currently available in other architectural notations. For enterprise modelling this notation should be considered as being at the *conceptual* level, while for projects it is more at the *contextual* level. This notation and approach is particularly aimed at collaborative discovery of business service architectures, which would then drive into a more complete architecture.

A new notation is required for this discovery phase as current notations detail the result of such an exercise rather than assisting with the discovery of the services themselves. Notations within Capgemini's IAF such as Business Activity have a similarity but are done after the services themselves are discovered. The purpose of the new notation is therefore to guide a collaborative exercise of service discovery which can then be formally documented within the current IAF[1] approaches of business services and business activities. For those familiar with IAF or other notations that also use the terms "what" or "why" it needs to be made clear that in the same way as with these architectural frameworks that a formal term is explained using information language the same approach will be used in this document.

It is also important to recognise that this paper aims to provide an approach for the discovery of services, particularly at the business level, but not their complete definition and implementation. Between discovery and definition of services to their final implementation is an extremely non-trivial process that must involve formal architecture, technical architecture and finally solution design. SOA is intended to be an approach that makes the IT estate better, this requires more formalisation rather than less in delivery, but enables that delivery to be done more effectively and accurately.

2. Executive Summary

Service Architecture, SOA, Event Driven Architecture, EDA and many other architectural buzzwords abound in technology at the moment. Although the standard view is that Services are a *technology* solution to add agility, past experience teaches us that technology solutions rarely deliver agility except when they are focused on the business visions. Services have been hijacked by technology vendors trying to sell integration and development tools, which most normally focus on "Business Process", "Orchestration" or "Web Services". This technology driven approach fundamentally misses the point of Services. The objective of a service is to represent *what* the business does and place a boundary which all parties, but predominately the business can agree on, it is this representation of the business which is the creation of a Service Architecture must be focused, technology is very much a secondary element. As ever with an IT related approach it becomes focused on technology rather than Method. The key to ***Service*** Oriented Architecture?

"It's the Services Stupid"

3. Abstract

Service Oriented Architecture is a powerful term that is regularly abused to refer to development technologies rather than an architectural approach, in the same was as Object Oriented Design was abused to refer to programming languages rather than a fundamentally different approach to design. This paper lays down a methodology for ***Service*** Oriented ***Architecture*** which deals only with Service as it applies to Architecture, and with Architecture where it is about exposing the Services

framework. Its purpose is to explain how a Service Architecture is created, how this in turn drives Service Orientation in broader Enterprise and Solution architectures, and how this impacts all aspects of IT delivery from Business Process to standard code development. The Service Architecture methodology described here is fundamentally a process of top-down discovery of what a business or business problem is about, rather than the "how" of implementation. Architecture is about context, frameworks, blueprints and standards, not about the individual aspects of delivery. Other methodologies already describe how to deliver software projects, this methodology helps provide the architecture to ensure that the delivery is Service Oriented.

4. Introduction

Service has become the accepted phrase to describe how systems should be exposed and co-ordinated. Building on the technology roadmap discussed in a previous paper[2] this paper concentrates on how this approach affects the approach to project delivery, and how a Services approach drives both enterprise architecture and business process efforts. Using a Service Architecture approach it becomes simpler to answer one of the most challenging questions in the delivery of IT - "what do I deliver where?". Where most methodologies, including RUP[3] and XP[4], are based around "single room" delivery approaches and concentrate on the artefacts within the project the key to modern delivery is that distribution of the project team is now the norm, not the exception. Industry efforts such as the OASIS SOA[5] group deal with an abstract notion for tools, others deal with the business view or a technology view. The objective of this paper is to provide a simple approach to the "Service" question in Service architecture and to provide a mechanism for planning, managing and delivering projects using SOA techniques. This paper does **not** cover the mechanism used to deliver an individual service, nor the project management methodology use, its intention is to explain how SOA can be used in conjunction with these approaches and how by driving SOA into an organisation and projects increases the flexibility of an IT organisation, and its delivery effectiveness.

5. Overview

This paper is explicitly only focused on the first element of either a business process, enterprise architecture, solution architecture or project engagement. The objective here is the first few days, and at most weeks, of a task to either map an enterprise's services, re-engineer its processes or to deliver an IT project for that organisation. As such this approach covers only

- Why Services need to be defined
- The importance of a common language
- How to discover what are the primary business services
- How to identify shared and supporting services
- How to define the interactions between services at a high level
- How to categorise services to help with management

This paper explicitly excludes

- Defining how processes work between services
- Full Enterprise or Solution Architecture
- The technical requirements of services
- The functional requirements of services
- The implementation of services
- Management of service programmes

The reason for this is that these other elements all augment the basic service model, it is therefore critical that that service model is done **before** all of these other elements are attempted. Otherwise the architecture cannot be said to be service oriented.

Figure 1 What, Who, Why, How

A service architecture follows a broadly four step process, or more accurately delivers three stages of the process and provides the direction for the fourth.

- **What**: Defining the scope of services, this about determine what the services actually are.
- **Who**: Who are the external actors that drive the services or with which the services interact.
- **Why**: Identifying why one service talks to another, and why external actors interact with the services
- **How**: The detail about the processes that co-ordinate the services and also the detail on how a service itself will be implemented.

This four step process is about getting focusing on the higher order elements first, which provides the context for the later stages. The Service architecture

- Defines the **What**
- Identifies the **Who**
- Highlights the **Why**
- Doesn't do the **How**

This means that future phases and approaches, whether Business Process, Enterprise or Solution Architecture will refine and detail each of these elements, but it doesn't alter the relationship. What drives who drives why drives how, and never the other way around.

Its critical to understand however that while this paper describes how a service architecture can be defined, it does *not* define how that service architecture can be delivered. It does not attempt to describe the full amount of information that is required for the successfully delivery of a services architecture solution or enterprise, this omission is deliberate. The purpose of this paper is to explain the start of the process and get things moving from the beginning of the roadmap. SOA does not replace other elements of architecture and delivery rigour, it just gives a proper framework into which they can sit. It's critical to understand that SOA provides only a framework, and that a complete architecture *must* deal with the **how** and provide a structure for strategy, implementation and support that is deliberately excluded from the Service definitions.

6. A common starting point

Architectural methodologies like Zachman[6] or IAF[7] start with the Context of the system or enterprise, its reason for existing and the intention of the efforts that are being undertaken. This is

a good start not just for an architecture, but also for business strategy. There have been several attempts at aligning business strategy to technology[8] but most have failed as the traceability from business to architecture to delivery to management has been at best weak and normally non-existent. For Service Architecture to succeed it must therefore be a representation that the business, who create the requirements, understand rather than a technology driven representation. By starting with this business representation, which is later augmented by architectural and technical services it is possible to create a clear, and auditable, trail from business strategy through IT Strategy and Enterprise Architecture to project implementations. This paper will therefore use the definition of service "*a discreet domain of control that contains a collection of tasks to achieve related goals"*[2] which can be used from both a business and technical perspective. A domain in this context is taken to mean a clearly identified area with a prescribed boundary. This paper does not describe an approach which replaces either architectural or business methodologies, but aims to provide a common language and interaction between all domains of the organisation or project. It also enables distributed delivery teams to be clear about what, they are going to deliver.

6.1 Creating the Big Picture

A major objective in undertaking a service architecture is to create the "big picture" this will act as an overall guide to an enterprise or project and provide a simple view of how the organisation, or project, splits its capabilities into services. This big picture is used to aid in understanding how change requests will be handled, new projects commissioned, and business change delivered through IT adherence. The big picture needs to be something that all parties agree on, and all parties use as a reference. For a project it might fit on a sheet of A4 or A3, for an enterprise it might be on A0. Make the big picture clear, use clear colours with a defined key, and most importantly ensure that it is kept up to date. The purpose of a good service architecture is not to focus people's minds on to the detail of the application, but to ensure they always have the context in which the detail is concerned. A service architecture concerns itself first with the "what", then the "why" and only finally the implementation question of "how". The Big picture is the one that tells all stakeholders the "what" and the "why", it is down to the various areas to determine the most effective "how".

7. Start at the top

The basis of this approach is to start at the top of the domain, whether this be at an enterprise level or for an individual area. The main reasons for doing this are

1. Organisations work "top-down"
2. Reduces clutter
3. Uses the organisational functions as its basis

Using the organisational structure or functions of a business as the basis for services is not a new idea[9] but these have mainly been process driven rather than service driven efforts and have attempted to answer all questions rather than aiming for clarity around the key question of what services need to be made available and how. It needs to be noted that while organisational functions tend to be relatively stable, the actual structure and departments can often be flexible especially when a new senior executive is appointed. The objective is to use the organisational functions, the "what" of the enterprise, and not the temporal representation of those within an organisation chart.

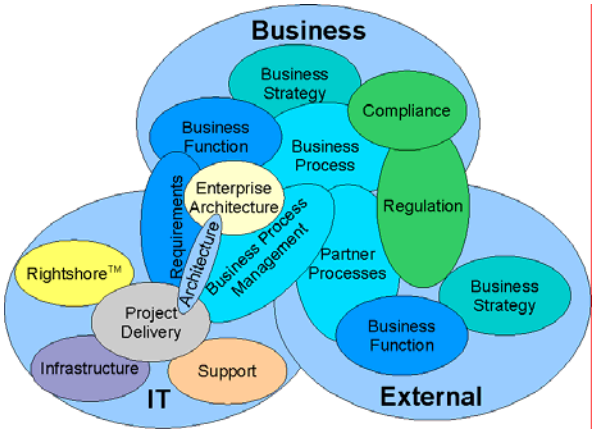

Figure 2 Standard View of IT and Business interactions

Figure 2 shows a standard view of how Business and IT interact and how External organisations are engaged. This map could be more complex than shown when each additional function is considered in greater depth. Elements like Testing for instance are included in the "Support" group which also includes call centres and application maintenance. The key in this diagram is that it deals with the skills that are being used and not the domain under which they are used. This is typical of IT driven solutions which most often concentrate immediately on the "How" of implementation rather than first considering the reasons for that implementation.

Service Architecture is driven by the domain, rather than the function and so represents a different mechanism for understanding how elements work together. The objective of this approach to service architecture is not to understand the skills elements that have commonality but the functional groupings of an organisation, and hence the services that it provides both internally and externally.

7.1 Why not start with Process?

A common approach historically to this sort of discovery work has been to use the high level business processes as the basis of discovery. This is explicitly *not* what this methodology attempts to do, this is for two distinct reasons:

1. Process based discovery tends to "drill-down" too early
2. Process based discovery tends to produce process "silos" of services and often fails to identify common ground between processes.

This service methodology is driven first by the broad "what" of the enterprise, and only second is processes considered as an orchestration of those "what"s. It's a normal understanding that business is driven first by its end-to-end processes, yet most organisations are structured around their key functions, and the end-to-end process goes between those functions. This suggests that organisations are, at a high level, first focused on the "what" and only secondly on the process, the "how".

Starting with process tends to lead to large "step through" workshops and interviews on that process, at this stage a service architecture is already imperilled as rather like the normal technical implementations, the "how" of the enterprise is considered too early.

8. Terminology

The approach described in this paper can be used at any layer of a market, an organisation or problem, the terminology and approach remains the same.

1. Level 0 - The start of a Service Architecture is the Level 0 picture. This Level 0 representation is the "70,000 ft" view of the domain under investigation. This could be to encompass a whole company, for instance to determine the 10 year roadmap for the organisation, or for a specific project.
2. Level 1 – Decomposition of the Level 0 model into finer grain services, this process can be continued to other levels (2,3,4 etc)

3. Support Services – Services which are not core to the general business or problem but which provide required functionality for the overall environment to function correctly. An example of this would be auditing for compliance.
4. Technical Services – Non business requirement functions that are needed for the IT system to be delivered. An example would be a hosting provider.
5. Contract – The functional and non-functional definition of a service, equates to a Service Level Agreement (SLA) builds on the work of Meyer[10].
6. Actor – A consumer (person, system or service) of facilities provided by a Service. Similar to a RUP Actor.

9. Collaborative Working

Key to creating service architectures is the process that is gone through to get the information and understanding. This is not a task that can be under-taken via a series of interviews with individual stakeholders, nor is it something that can be gleaned from documents or websites within the company. It is all about creating a common dialogue between the various different groups and deciding upon a boundaries that work across the business. For this reason the best way to get a decent services architecture is to engage in a set of Collaborative working, an area so well documented it has its own conference[11], what is recommended for either the project or enterprise service architecture is to start with an intensive session, between 1 and 3 days depending on the scale of the problem. This event needs to be well prepared with all of the required stakeholders in place, and information available on which to make decisions. Mostly what is needed however is the people who work inside, and outside the company who understand how the business, or project, functions. Creating the service architecture is about creating the broad sweeping pictures, not the detail within those images, as such ensure you don't get bogged down with a point of detail on a specific flow, as long as everyone agrees it goes from "Service A to Service B".

A properly facilitated event will create an invaluable resource that will deliver its benefit many times over. And by getting everyone together you ensure that everyone is agreed on what the big picture is. This picture gives you the map for the enterprise or project and enables further analysis and detail to be created, while ensuring that these efforts are not impacting each other. If an organisation is serious about Service Architecture it will need to either partner with an organisation that runs these sorts of events, or create the infrastructure itself. Once the decision has been made for the first time the architecture becomes a living artefact that needs a regular review. Reviews should be initiated as part of normal business and technology change procedures, it should become enshrined in these procedures to first consider the impact on the service architecture. In the absence of such a review it is recommended that the follow minimum review periods be adopted.

- Level 0 this should be done once a year,
- Level 1+ once a quarter

A review both as part of a business or technical change, or via a standard review period, should not attempt to re-create the whole architecture but focus on those changes that may impact parts of the architecture, the objective in these reviews should be to understand the change, not to re-create the architecture.

These created artefacts must therefore be available not via word documents but via a collaborative environment which people can quickly access, and potentially augment. Without an environment in which the big picture is created, agreed upon and made visible a service architecture will become yet another piece of "shelfware" that means nothing to anyone. The approaches, and terms, used in the Service Architecture must be driven into other areas, for instance the Enterprise and Business Architectures, to ensure there are solutions that can be completely traced.

9.1 Coping with Major change

If a review identifies fundamental changes in the way an enterprise operates, most often due to a large scale acquisition, merger, disposal or outsource, then a full architecture review should be undertaken and potentially a new top down architecture created. This should be enshrined within these business change programmes, and indeed form the basis for how the IT organisation is to be guided and directed by the business. The intention of such a review within a major change programme is to ensure that programme's effective implementation; it should therefore run in parallel with such efforts. Comparing the established service architectures of organisations that are attempting an acquisition or merger is a good way of noting the compatibility of those organisations. While they may external exhibit the same functions to the market a service architecture comparison can identify potentially serious discrepancies in how the organisations approach that common task. This matching of service architecture also represents a good way of identifying areas of commonality that can be turned into single shared services to deliver the expected cost savings.

9.2 Language to determine services

When determining a service architecture a useful technique is have people imagine they are looking at the enterprise, department or project from the outside. At this level what do they see? What a Service Architecture should be driving them towards is identifying the types of work that are being undertaken. "What does it do?" is the driving question at this level, the objective is to understand the form of the services rather than their details, so as the discussion delves down towards "well I ask for a paper clip and they give me a quote, then I submit the purchase request and they….." it is important to ask for a term that groups all of those elements together, ask "So what would you call that type of function in your company?", or "and as a group what are they know as?". It is critical not to get bogged down in the process elements and to be thinking in the service architecture purely of the groupings. Once the primary groupings have been defined you can then work on the primary tasks, again doing these at a high level, the intents, rather than the specific elements that are undertaken.

10. Creating a Service Architecture

In order to explain how a Service Architecture is created two different scenarios will be used. The first will be to describe the entire service architecture for an organisation, the second is for a specific project within that organisation. The organisation "Oblivion Widgets Inc" manufactures widgets, and the project is to enable vendor managed inventory (VMI) of its stock.

10.1 Template of a Service Definition

The following is an example of the sort of information that needs to be captured during the service definition. As can be seen from this example its relatively high-level and concentrates on the drivers for requirements rather than the full detail. This is done deliberately to avoid going deep to early, and to ensure that the broad communication elements are clear to everyone. By identifying the different, often competing priorities, of the key stakeholders and actors it helps to identify the actually important elements, and the potential issues that could arise.

10.1.1 Logistics and Warehouse

The Logistics and Warehouse Service is concerned with the allocation of products to vendors and the replenishment of warehouse stock to meet anticipated demand.

10.1.1.1 Business Owner(s)

Table 1 Logistics and Warehouse Business Owners

Name	*Role*	*Description*	*Priority*
Brian	Head of Supply Chain	Responsible for all of the business and technical elements related to the supply chain infrastructure, including Warehouses, Logistics and stock forecasting.	Minimising of stock while meeting customer SLAs.

10.1.1.2 Actors

Table 2 Logistics and Warehouse - Actors

Name	*Role*	*Description*	*Priority*
Customer	Organisation that buys, and potentially sells on, widgets	Typically medium to high scale retailing operations	Stock availability and low prices
Supplier	Manufacture or Wholesaler of component parts needed to make widgets	Typically Asia Pacific companies for large bulk elements and some small European companies for specialisation	High Demand, prompt payment
Logistics Company	Used when the standard supply chain cannot cope with local demand, or to supply to global markets	A combination of local haulage firms and international shipping and logistics companies	Demand, load size.

10.1.1.3 Primary Tasks

Table 3 Logistics and Warehouse - Primary Tasks

Task	*Description*	*Pre-condition*	*Post-condition*	*Invariant*
Ship Order	Request for an order to be shipped.	Goods on the order are available	Order is assembled ready for delivery, delivery is scheduled with the client.	Price of Goods
Add to Stock	Add new items into stock	Item is available and is a valid and known product	Items are added to the stock count	Stock of items un-related to the new items
Deliver	Request an external company, or internal logistics, to supply an order to a customer	Goods available for shipping	Goods removed from warehouse and placed on appropriate delivery route	Items on order
Supply	Receive supplies	Order had been placed with supplier, or supplier is managing own inventory	Inventory of supplied item is increased by amount supplied	Stock of items not related to this supply order

10.2 Level 0

The key when considering architecture at Level 0 is that each of these services must be core and central to the actual *business* being considered. For this reason support services, which may have large departments, would not be considered as Level 0 services. At Level 0 the important element to consider is that each Level 0 system could potentially be considered as an area in its own right, so its replacement would have minimal impact on the other services in the domain. For an organisation this often means that the model reflects areas that could be either outsourced, sold or partnered. And for a project it often represents the different business objectives that the system has. The other key element in deciding what the Level 0 services are is that *combining level 0 services into larger domains would not reduce the high-level clarity of the system.* As a rule of thumb the number of Level 0 services should be between 1 and 5.

10.2.1 Enterprise Level 0 model

For the enterprise the approach to understanding the services in the Level 0 model is to understand **what** the key business areas are that make up that enterprise. With Oblivion Widgets these can be split into four key areas, Sales, Manufacture, Logistics & Warehouse and Finance, these represent the central elements of the business. The key actors at this stage are those that interact with the services externally, not the internal actors who provide that service. The question therefore is "what is it?".

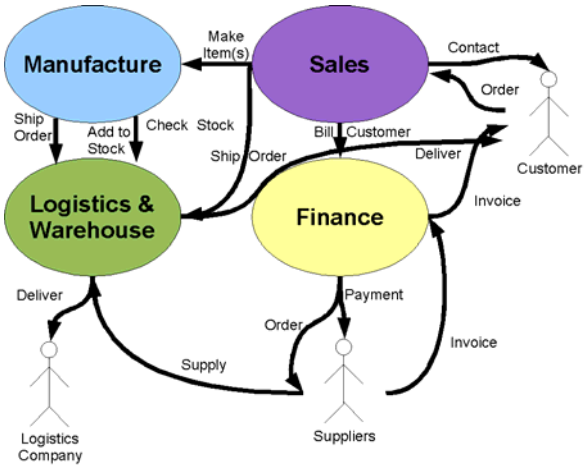

Figure 3 Enterprise Level 0

Figure 3 shows the level 0 model for Oblivion Widget with the key external actors. It also details the primary interactions that those actors have with the services. These are **not** individual functions, but descriptions of the purpose of the interaction. On this diagram there is no description of the process of the interaction, only the start, or end, of an interaction. This picture is deliberately simple, and should be kept so. Its intention is to act as the reference point for all initial decisions within the organisation and as a reference point to which every stakeholder, business, IT, internal or external can agree. These boundaries also mark the areas within which change will be managed and constrained.

10.2.2 Project Level 0 model

Within a specific project exactly the same approach is taken, this time at a much lower level, if an organisation has undertaken an enterprise Service decomposition then its important for this to be used as the base for the project, not only does it represent a head-start on the process but also ensures that the required stakeholders are aware of the project and its impacts. The first element is to understand what is required of the project as a simple statement. The objective here is to minimise the amount of stock carried while maintaining, or reducing, the amount of out-of-stock issues that customers have.

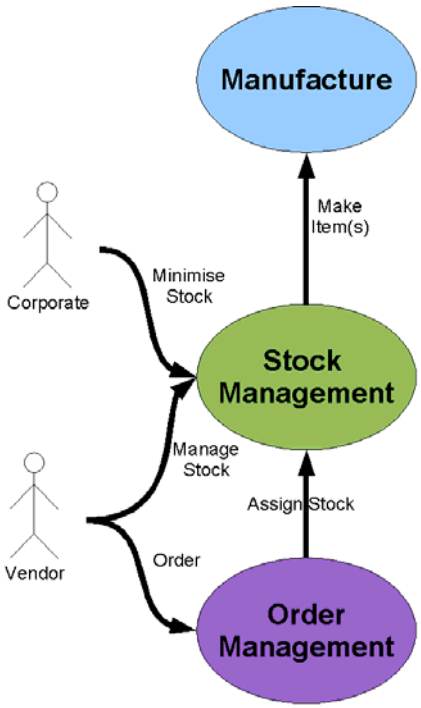

Figure 4 Project Level 0

The project level 0 (Figure 4) is equally simple, and hopefully equally powerful, the intention is to understand the objective of the project from a simple diagram and the domains which the project covers. From a top level the objective of the project is therefore simple, the organisation (Corporate actor) wishes to minimise the amount of stock required, and the vendor is now responsible for managing their own stock levels. Flows which are not directly changed in the project scope, e.g. ship against order. Are excluded at this level.

In order to demonstrate the purpose and options at this stage it is recommended to create a number of business activity diagrams, these should be seen not as formal process definitions, but as sketches that represent either the different options or the different scenarios. Figure 5 and Figure 6 show two such diagrams. These are very high level and are used to demonstrate how a polling (former) or event (later) approach to VMI would work.

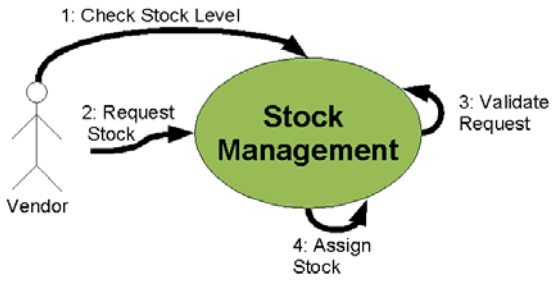

Figure 5 Standard Business Activity

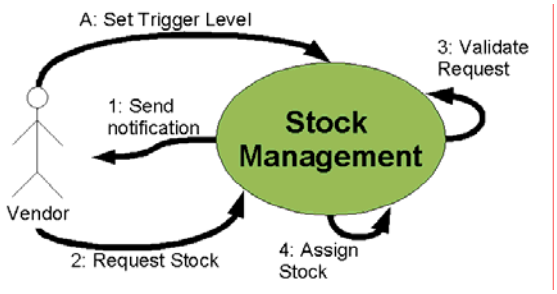

Figure 6 Asynchronous Business Activity

The objective here is simplicity, **not** to go into the detail. Level 0 should be kept deliberately abstract and be clearly understood by all stakeholders. If understanding of the domain is lost at this stage there is little hope that the project will succeed. It is normal that the activity diagrams used here are revisited in detail by either the business process or architecture streams of a project or programme. It is important to retain the *context* described at this level, even if the full content is much more detailed.

10.3 Drilling down to Level 1

Level 1 Services are where element start to become more "real", quite often these can be identified not just as areas in which people work, or conceptual goals of a project, but as the actual day-to-day areas in which people work, and the IT services that will be implemented to support them, at an enterprise level these may map to the departments in the business, and within a project to the roles and IT Application areas that are required.

There are several elements that need to be borne in mind when considering decomposition. The first is that the enclosing service confers behaviour and management onto those at a lower level. This means that while services from two distinct domains can interact, they have to do so in a manner that enforces the surrounding, as well as service specific, contract. This inheritance of principles, contracts and other functional and non-functional attributes is central to a well constructed model. This rather than thinking of a circle on a circle, a better mental model is to think about nested spheres (Figure 7).

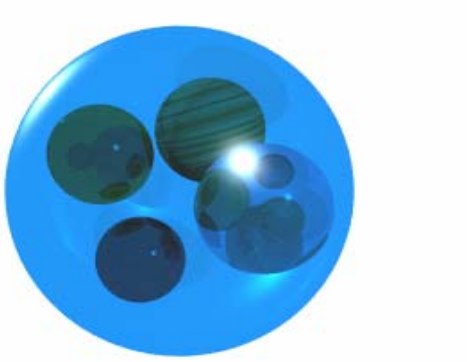

Figure 7 Services nested inside services

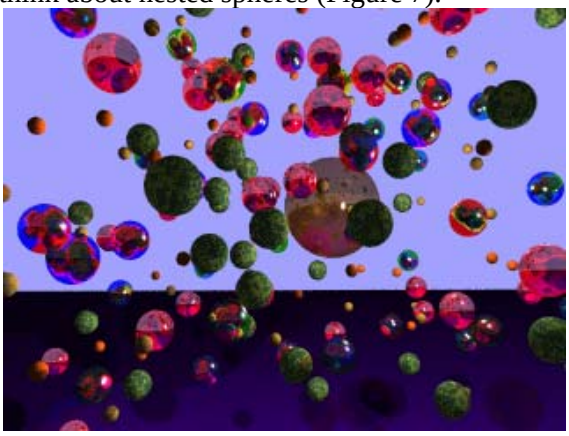

Figure 8 Too many Services, too little hierarchy

The second is that the purpose of this classification is to enable simple navigation of the service architecture to ensure it can be understood and future requirements can be easily classified within the service models. The purpose here is to avoid the "death by availability" challenge that comes with too flat a structure. Without a defined structure and governance model it becomes impossible to manage the complexity of the enterprise (Figure 8). Its this problem that SOA tries to address so its important that any Service model is designed to aid in comprehension, not to stress the complexity of the environment.

10.3.1 Enterprise Level 1

Rather than decompose all of the Level 0 services one service has been chosen, namely manufacturing. The principles remain the same

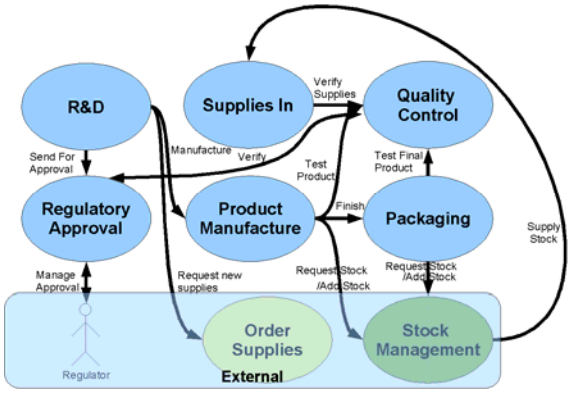

Figure 9 Enterprise Level 1 for manufacturing

The Level 1 for manufacturing splits down Oblivion into six distinct areas. Thanks to the centralisation of purchasing around stock management the only area that needs to make direct purchasing requests is R&D as they use newer materials than are currently being used. A key part of this level 1 diagram is that it identifies other level 1 services with which it interacts. This is why some form of common tooling can help in creating the service model so changes in terms are instantly reflected across the various instances of that service.

An enterprise Level 1 sometimes matches the operating departments within a division, or at least a potential set of operating departments. It is risky to attempt a one-to-one mapping of services to departments as these can be subject to change, while the organisational functions they undertake will remain relatively complex. The importance of a service is that it represents "what", the department often represents "how" . This is quite often the stage at which more discussion is

required as the key is to understand the key services that the division is offering, rather than just repeating that "Frank runs both Packaging and R&D", if the two elements have a clearly distinct purpose then they should be represented as separate services. Again logical groups should be represented as one service and further decomposed as required.

10.3.2 Project Level 1

A project level 1 is often the stage at which the requirements become more apparent for the system, where the Level 0 (Figure 4) details the intent of the project and sets it context, the Level 1 indicates the actual work to be undertaken and explains how it will work.

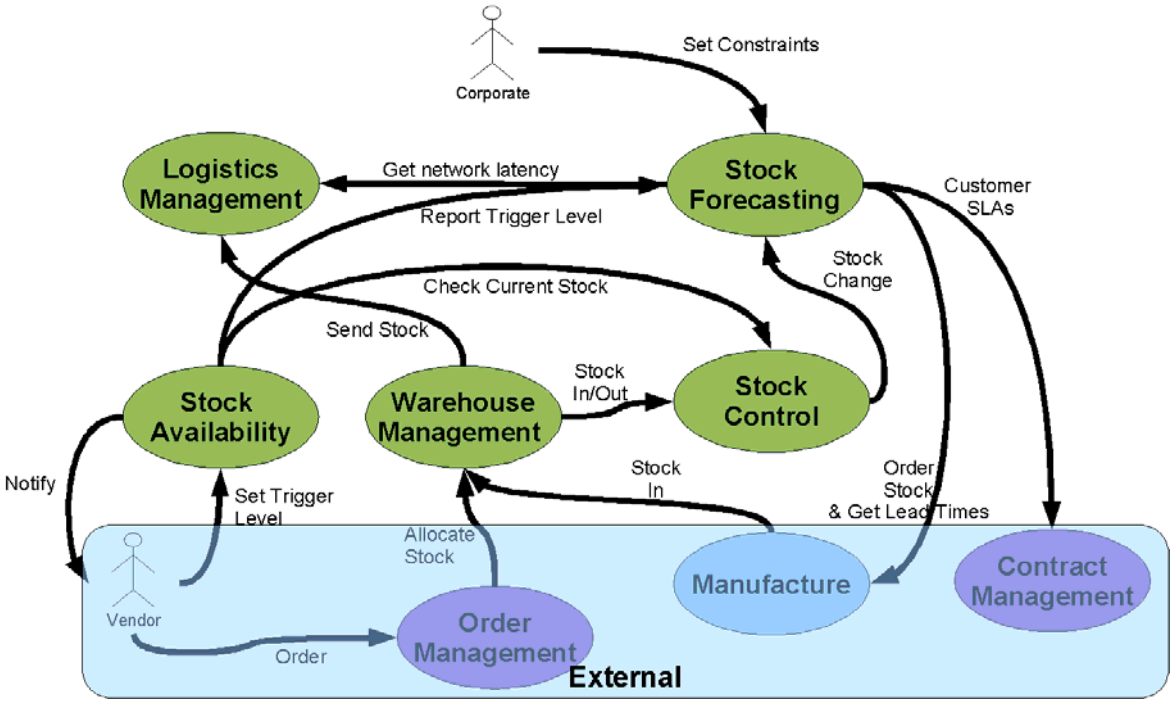

Figure 10 Project Level 1 for Stock Management

Figure 10 applies the same principles used to create Figure 9, only this time applied to the specific requirements of the project. Again the external interactions are reported, and the major interactions within the service model. Level 1 often refers to the major software components of a delivered system, these can be as large as whole packaged solutions, for instance Logistics Management in this case, or bespoke areas of targeted development, for example forecasting. It is within Services at this level at the requirements are often initially captured. For small projects a Level 1 decomposition will be all that is required. Level 1 can also be more complex than level 0, but a maximum of 8 Level 1 services for each Level 0, with a normal amount being around 4, should be used as a guide.

10.4 Further refinement, virtual, support and shared services

Further refinement can be driven by two different objectives. The first objective is to delve deeper and understand the problem domain more. This will take a similar mechanism as that defined for Level 0 to Level 1 decomposition and produces the Level 2+ decompositions that might be required. The other refinement is, for a given service, to focus on the different external representations that it may have. This is used for Services, and collections of Services, which have a number of external consumers which interact with a common set of functionality in differing ways. This should not be used where a service just has multiple actors who call differing

functionality, for example a purchasing service where some people buy, some price, and others approve, these functions make up common areas defined by roles within one services.

10.4.1 Virtual Services

Virtual Services should be used where a collection of internal services is combined to provide an external view for a customer, thus creating a "virtual service" that is one which provides no direct business function but which offers a façade over those services. It should also be used where one service provides distinct operational objectives depending on how it is being invoked.

Virtual Services are often the information or interaction points in the systems. Because they are virtual, have no direct business domain, does not mean they are trivial or are not owned by a clear domain within the business, and often can be the source of projects on their own within Enterprise models. Sales or customer portals are often represented as virtual services and owned by the Sales Level 0 service.

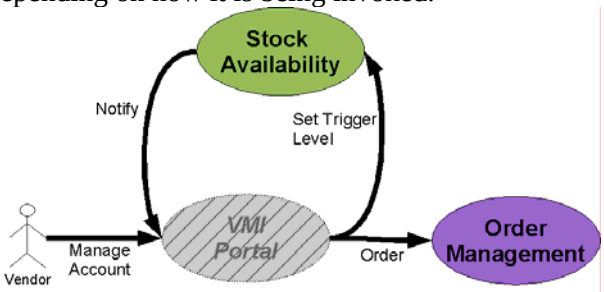

Figure 11 Virtual Service Example

How they are differentiated from normally services, beyond being represented hatched on a diagram, is that they are not a service in which business logic will be implemented. Virtual Services therefore provide a way to indicate where business logic can be co-ordinated and potentially simplified, but the implementation of the actual logic should only be done in full services.

10.4.2 Support Services

Support Services are often the technology elements that the business doesn't care exist, as long as they exist. These are split into two groups, those where have a clear encompassing service in the enterprise model, and those which fall within Shared Services. Support Services have two distinct groups, technical and associated.

10.4.2.1 Technical Support Services

Technical Support Services can vary from elements such as hosting and printing, through to specific plant control elements or an RF-ID portal. As with other services it is possible for these to be relatively high level and be decomposed into further technical services. The key element with technical services is that they provide support to a business function, rather than being the specific business function themselves. For this reason it is important to differentiate between pure technical services, and those business elements that have been automated. It is also critical not to fall into the trap of creating services from existing technical elements just because they exist in a given form.

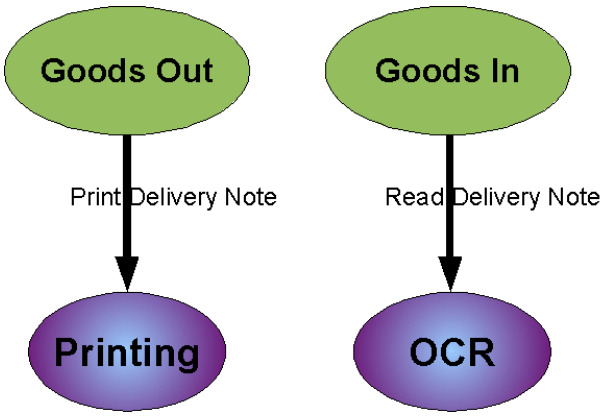

Figure 12 Technical Support Services

Technical support services are those elements that support the business, not business functions in their own right. Having an ERP system does not mean that "Enterprise Resource Planning" is a specific business domain in its own right. Nor does the existence of a CRM system mean that "Customer Relationship Management" becomes a support service automatically. The objective of a service architecture is not to determine the technology that will be used, but the domains of functionality that are required.

It is the job of application architecture and design to determine the most appropriate technologies and their implementations. Technical Support Services therefore are used only to indicate those elements which indicate a category of supporting function that is required by the main business elements. These are normally only defined at lower levels of granularity, and are normally shared between multiple domains, they are also consumed by Business Services, rather than being consumers of business services themselves.

10.4.2.2 Associated Support Services

Associated Support Services are those elements that are not required for the business operation of the system, but which are required for the business to operate. Elements such as Human Resources, Desktop Support and other internal only functions of business, and projects, need to be represented on the overall service map. It is important however to remember that these elements are **not** central to the operation of a business and should not therefore be exposed as Level 0 services.

Associated Support Services should always be represented on diagrams in a specific way in order that everyone is clear as to what the primary goals of the project or enterprise are. Associated support services may often be important to the operation of the enterprise or project, but their existence is only to ensure that the business services are delivered, without the business services there is no reason to have an associated support service. Associated Support services are often also Shared Services that provide a supporting function to multiple business services.

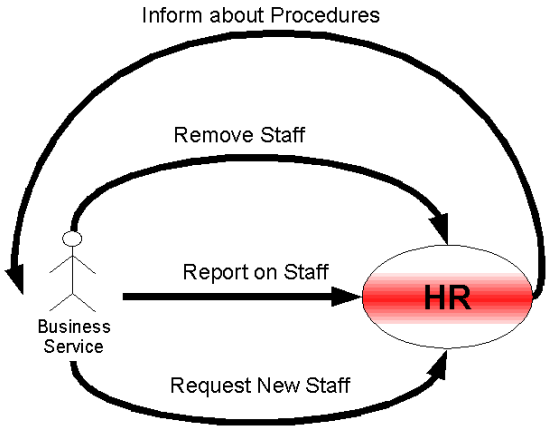

Figure 13 Associated Support Services

The relationships between associated support functions are often irrelevant at the business level and just need to be described only on the Associated Support service.

10.4.3 Shared Services

As a service architecture drives into more detail it is common to identify certain services that are common between multiple business areas. These services maybe technical or support services, or may on certain occasions be clear business services which are defined to work in two places. In all

of these cases it is important to recognise that some elements of the service might be shared, or that the whole service is common between multiple areas. Representing shared services requires an increased level of control and visibility, it is recommended that shared services are grouped into two groups:

- Technical and Support services split by
  - Shared between multiple business services at all levels and across Level 0 service boundaries
  - Shared between multiple business services with a specific level of a hierarchy (e.g. within the Level 1 diagram for a Level 0)
  - Those with common or similar bases, but differing drivers or implementations.
- Business Services split by
  - Totally shared services with defined business reason for being shared
  - "Apparently" shared service, these are services that appear to have the same characteristics but are deliberately separated
  - "Common Base" these are business services that shared a common base of context but have been specialised for a particular business purpose

The reason for making these classifications is that it enables IT teams to understand which services **can** be re-used across domains and which *even though they appear common* should either be hosted or implemented separately due to differing business demands. It also helps to drive both business and IT investment decisions about where the most benefit could be derived, and to help with the prioritisation of work. It is important to consider that Services with differing objectives and measures can be implemented as a shared service by building the service to meet the most stringent requirements. This assessment should be made on cost-benefit grounds and requires a realistic view of possible re-use. This can either be done at this stage, or more normally identified as a potential at this stage and then clarified during a future iteration. The advantage of a clear Service Architecture is that it gives clarity and context for that decision.

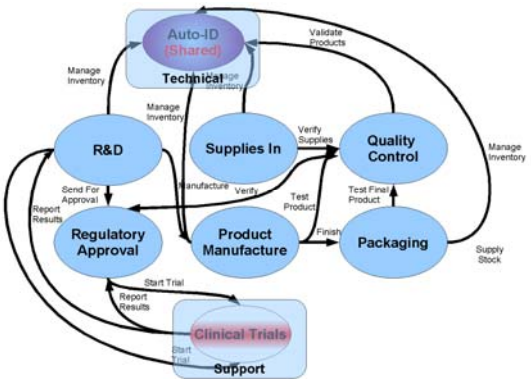

Figure 14 Cross Domain Technical Service and Domain Shared Support Service

A key part of these diagrams however is that they direct the strategy of these support services. If an organisation wished to trial multiple different Auto-ID solutions it would be reasonable to specify these services as not shared, but as having a common base. A future strategy that adopts one of these approaches would then be focused on unifying these disparate services. Figure 15 gives an example of another secondary diagram that can be used to represent common bases for services, as with a multiple pilot approach for Auto-ID all of the various types of forecasting are at their heart forms of demand forecast. This does not mean that these services are the same, or even that they share any business drivers or goals, but that the conceptual frameworks for the services share a similar base.

Figure 14, which is Figure 9 without the external services, shows an example of both a cross domain shared service and a domain specific share support service. The examples chosen here are for Auto-ID (barcodes, RF-ID, EPC etc) technical service, which need to be available across the organisation, and of clinical trials support services which while specific to this domain, are to be shared between the R&D and Regulatory efforts. Diagrams such as Figure 14 should be used as decorations to the key service models to demonstrate additional elements, rather than trying to put all these elements on the key business diagrams

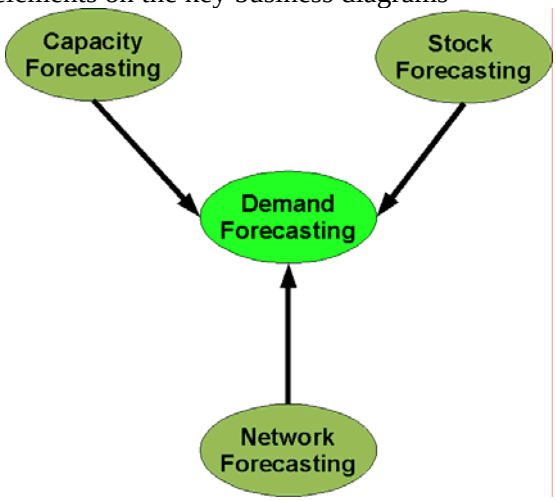

Figure 15 Common Base for multiple Services

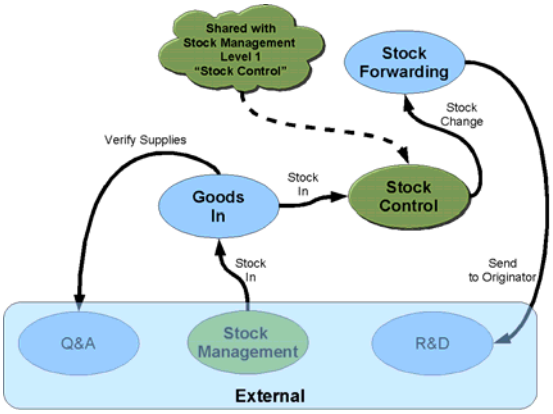

Figure 16 Shared Business Service

It is important to identify these common bases as they can, on occasion, lead to the identification of potential shared services and also identify areas where business services can co-operate on common ideas and approaches. This is particularly important when trying to foster cross-functional links in organisations as it can be used as a basis for common understanding. Figure 16 shows an example of a shared service, in this case explicitly referenced, this is an example of where exactly the same functions are duplicated between two parts of an organisation, but the processes, priorities and directions of the service remain the same.

These shared services can often be physically manifest by separate departments, but which a shared IT and audit capability. It is important to note at this level however that they are, from a *business* perspective performing the same function and that any real-world separation is a matter of logistics or convenience not of differing business objectives. Figure 17 details a case where services, though apparently shared, are in fact distinct services with distinct objectives.

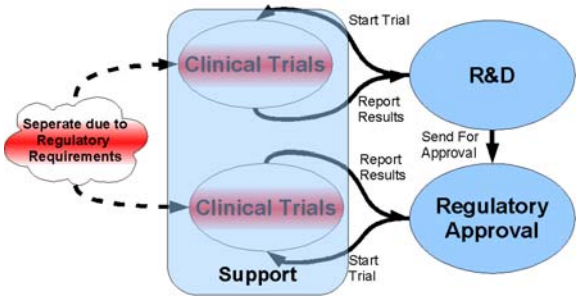

Figure 17 "Apparently" Shared Services

In the case of Figure 17 this is due to regulatory pressures on the market in question which ensure a clear separation between clinical trials during the course of a companies R&D and those conducted for regulatory approval. "Apparently" shared services are often driven by regulatory drivers, either due to differing legal rules across regions or due to specific legislation around a business domain. When decomposing these apparently shared services it is normal to identify elements that are shared services, and others which must remain separate. The difference between these three forms of sharing, Common Base (Figure 15), Shared (Figure 16) and Apparently Shared (Figure 17) is more than just a semantic one. Common Base services are functionality different but have either a common "ethos" or core approach, they however are not the same either in form of their objectives or their direct function. Shared services are alike in both their objective and their function and can be view as a **single** service shared across an organisation. Apparently Shared services are those which share both objective and function, but are separated due to indirect concerns, in other words they are kept separate by factors not driven by the business. By identifying these groups, normally as a third or forth iteration after the basic business service model is done, businesses and IT can identify opportunities for consolidation and co-operation, but done within the correct legal and operating constraints.

11. Creation of a Service Architecture

Having been through the various different types of services, the decomposition of those services, the addition of support and shared services and the categorisation of all of these elements it is clear that if this was all attempted in one go it would become needlessly confusing. This section therefore deals with the standard approach, and timelines, used to create a service architecture, and the deliverables at each of the stages.

11.1 Stage 0 – Pre-work

Before engaging on a services architecture it is important to have a broad understanding of the area for which the task is being under-taken. If undertaking the task at an enterprise level this should include the external drivers on the company as well as an understanding of the sector it is in. If it is a specific project or programme then it is key to understand the primary drivers for the project. The result of this pre-work should be for the analyst, architect, consultant or manager to understand the external drivers for the architecture that is being created. A key element to start at this stage is the glossary of terms, so certain elements can be quickly referenced when required.
At this stage it is also important to identify the key stakeholders who are required to create the Level 0 and Level 1 service architecture, and plan the event to which they will be brought in order to create the architecture. This event must be properly facilitated with all information gathered at the event collected. During the Pre-work the initial collaborative tools for sharing the outputs of the work should also be created, this may change over time but a lightweight first attempt is a necessity to ensure the architecture becomes a living thing rather than a musty document.

Table 4 Stage 0 Deliverables

Deliverable	Description	Enterprise Duration	Project Duration
Sponsorship	Without clear executive sponsorship for the effort a Service Architecture is doomed to fail.	N/A if this isn't achieved Stage 0 hasn't been reached.	N/A if this isn't achieved Stage 0 hasn't been reached.
Stakeholder Identification	The key stakeholders from the business, technology and ideally external interactions need to be identified and contacted for participation in the event.	3 days to 2 weeks depending on scale and external involvement.	Normally takes around 3 days to identify the people.
Event Planning	The Service Architecture event needs to be scoped and planned. This requires a certain amount of pre-planning to ensure any information required at the actual event will be available. This also involves managing the stakeholder attendance and flagging any issues with non-attendance.	2 weeks to 6 depending on the scale and number of participants.	Between 1 and 3 weeks depending on the scale of the project and the logistics involved with getting everyone together.
Technical Support	The collaborative tools and information availability to support the event	1 week to set up	1 week to set up
Information Collection	Gathering of the specific information to support the event	2-6 weeks depending on the depth and amount required	1 to 3 weeks depending on the depth and amount required.
Statement of Context	The mission statement for the event	2 days with event sponsor	1 day with event sponsor.

An example project plan for this is shown in Figure 18, this is typical of the planning that needs to go into a services architecture event for a reasonably sized project. This would be sufficient preparation work for the sort of project that is expect to take between 9 and 12 months in duration and involve minimal external interactions within its service architecture.

ID	Task Name	Duration
1	**Stage 0**	**9 days**
2	Sponsorship	0 days
3	Stakeholder Identification	3 days
4	Event Planning	7 days
5	Technical Support	5 days
6	Information Collection	7 days
7	Statement of Context	1 day

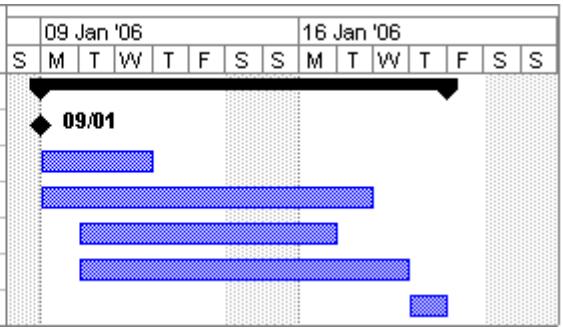

Figure 18 Example Stage 0 project plan

At this stage the intention should be to keep the team small at this stage this normally means

- Project Manager for delivery
- Lead Architect
- Lead Business Analyst
- Administrative assistance (P/T)
- Technical Support (P/T)
- Event co-ordinator (P/T)

In an organisation set up to do these events on a regular basis it is normal for the last three roles are part of the physical event venue team.

For a large enterprise definition exercises the key driver in duration is the number of stakeholders who have to be corralled together, this is where clear executive sponsorship with the ability to pull people into line is critical. The team to do the work should not be much greater than that for a normal project, normally there is an addition of one analyst and a supporting architect if the level of work is particularly large. The key at this stage is that it is about comprehension and not about gathering of requirements. Operational Research groups are of most use during an enterprise definition stage 0 to obtain the required level of relevant information.

At this stage there is no service architecture.

11.2 The Event

The objective of the event is to get all of the stakeholders to agree on their position. This means it must be run as a collaborative event with all stakeholders encouraged to view their opinions. A properly planned event, and the supporting infrastructure, should not be mistaken for a "workshop". The intention here is to produce a specific deliverable, paper is not required, whiteboards are. Tables should only be used for breakout sessions and the event should be scheduled to run 100% of the time during the day with moving refreshments available all the time. The event venue infrastructure must have access to the internet, all elements must be captured and available via a collaborative site by the end of each day. Some basic rules must also apply:

1. No mobiles
2. No email
3. Timetable at the start of each day is fixed, end means end.

11.2.1 The Facilitator

At the event facilitation is key to the successful creation of the architecture. The facilitator requires the following skills:

- Strong Communication skills
- Strong listening skills

- A broad understanding of the business domain
- The ability to co-ordinate teams
- The ability to end discussions with an agreement

It should be noted that none of these are technical skills, and indeed it is the case that service architecture can be created without having an architect undertaking the facilitation role. A business consultant or business manager is as equally well equipped to undertake this process if they have the right level of understanding. It is most normal however that this be an Enterprise Architect who can bridge the gap between the Business and IT functions and work with the various different elements on implementation and strategy.

The final and most important skill in the facilitator is that *they have done it before*. This means that a facilitator should not be undertaking an enterprise wide service architecture discovery if they have not previously done the task either on large programmes or on organisations of similar or slightly smaller size and complexity. A facilitator for a project should have previously undertaken a similar role, or been an assistant in a large programme or enterprise wide mapping exercise.

When determining what is a service and what isn't, there is no substitute for experience.

11.2.2 The Level 0 picture

The focus for level zero is the same question as for all levels "What is it?" the ambition here is first to get the high-level picture. Section 10.2 describes what is required at the end of the process. The first, and most important element, is to determine what the actual services are, without worrying about the interactions. At this stage the most valuable tool is a white board as there may be many candidates for Level 0 services depending on either the perspective or opinion. The target is a clear statement of *just* the Level 0 services.

For the Oblivion Widgets Inc Enterprise the first element is going to be the core Services them selves. This just describes the "what" of the enterprise, and absolutely no more. It is critical that a business agrees on this diagram before attempting anything else, because if the top can't be agreed upon there are some significant differences in how the organisation perceives itself. For Oblivion Widgets Inc that diagram is shown in Figure 19 and it represents on the four key areas that define what the business is about.

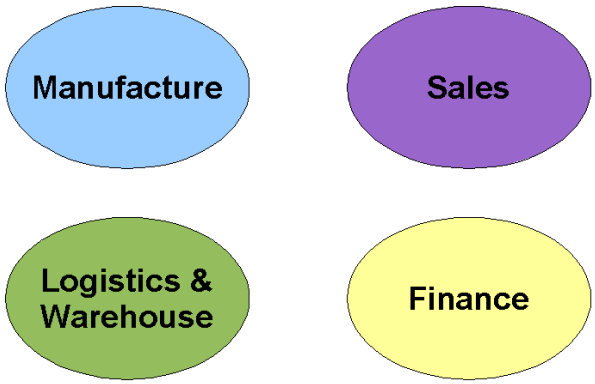

Figure 19 Initial Enterprise Level 0

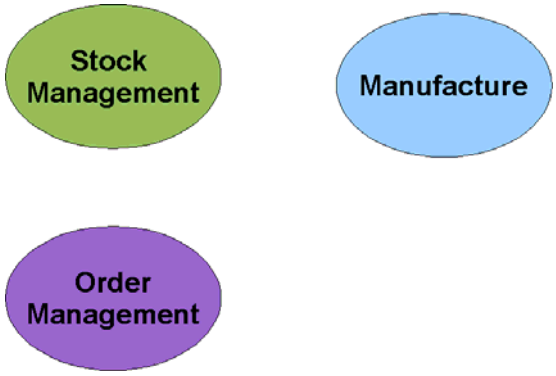

Figure 20 Initial Project Level 0

For a project the approach, and need for agreement is no different. The question is "what is it about" and the answer the key domains that need to work together to solve the problem. Figure 20 shows this minimal set of services required for the vendor managed inventory project. It's important to note on a project basis that the driver is still what are the *business* focused rather than project focused services that are required. The services created **must** fit within the context of the overall business and not just be specific to the project.

The level 0 picture in Figure 19 represents the most powerful picture for an enterprise, it shows clearly *what* the organisation is about. For a project the type of Level 0 diagram shown in Figure 20 can sometimes be compelling but more often the next level of refinement is need to really explain what the *project* as opposed to the services it delivers is about. This is because a project has explicit drivers that provide context to the services, while an enterprise *is* the services themselves.

11.2.3 Adding the Actors

The next elements to be added to the diagram are the external actors these round out the domain under consideration and represent the totality of the "what" within the Level 0 model. The key here is that the actors are *external* to the services, rather than being the internal objectives of those services.

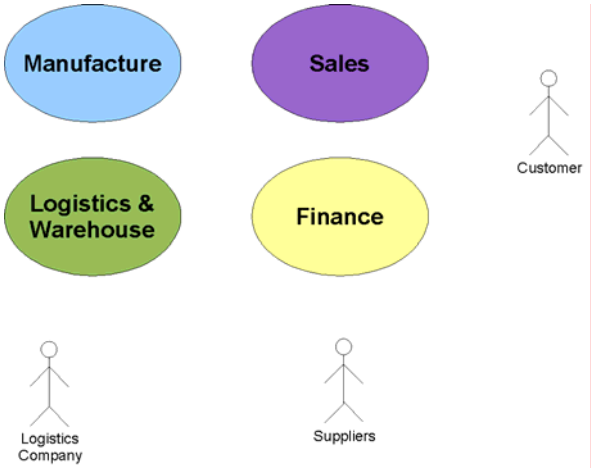

Figure 21 Enterprise Level 0 with Actors

.

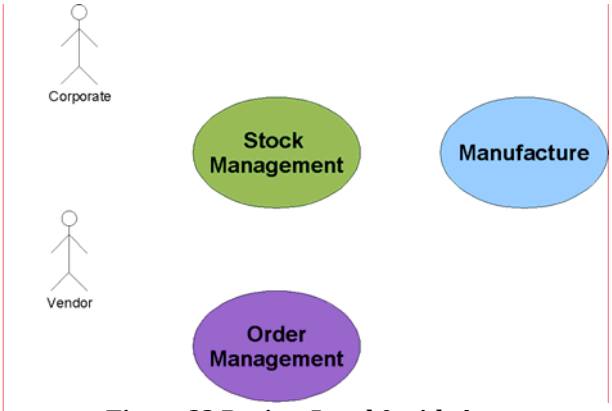

Figure 22 Project Level 0 with Actors

For an enterprise this (Figure 21) expands on the "what" of the enterprise to talk about the "with whom". This can be a powerful tool to help and enterprises determine the scope and importance of partners. If they are not important at Level 0 they should probably not be considered strategic. For Projects (Figure 22) the impact can be less as although it defines the totality of "what" for the project it doesn't at this stage define the full "why" of the project. It is therefore important to understand at what level the diagrams become truly useful. For our Enterprise it was reached in stage 1, for our project we have to refine again.

11.2.4 Adding the interactions, adding the "why"

The next element in creating the Level 0 picture is to understand *why* various services and actors interact. This is the key question for the facilitator to pose at this stage, "does service X interact with service Y and if so why does it interact". The aim here is to start defining the value contract for the service that it offers to the world. This then delivers for the enterprise, Figure 3, a full definition of both what the organisation does, and why it does it, and for the project, Figure 4, completes the picture as to why the project is being under-taken.

During this the facilitator must continue to consider which single diagram will best convey to all stakeholders most powerfully what the service architecture is about. For a project it is normally the element at this stage of refinement, but often for an enterprise the first iteration can often be the most effective due to its absolute simplicity.

11.2.5 Fleshing out the services

Once the full diagram has been defined its time to fill in the information required to describe the service. The sort of information required is described in the Template of a Service Definition section. It is recommended at this stage that break-out groups are created to focus on each of the services; these should include people from that service domain, and people from the services that interact with that service. These groups should then report back the definitions for a final confirmation by the whole group.

11.2.6 Level 0 Deliverables

At this stage the following elements should have been created

Table 5 Level 0 Event Deliverables

Deliverable	Description	Enterprise Duration	Project Duration
Service Diagram	Diagram of the services at this level	½ day	2 hours
Actors	Identification of the key external actors	2 hours	1 hour
Service/Actor Interactions	Identification of the interactions between services and actors	2 hours	1 hour
Service Definition	Definition of the services and their interactions in more detail	4 hours	2 hours
Report out	Confirmation of all findings	30 minutes	15 minutes

11.2.7 Drilling down

Drilling down from the Level 0 into the lower level elements is just a series of repetitions of the steps described in sections 11.2.2 to 11.2.5 and creating the deliverables described in section 11.2.6, but applied to the lower level. It is advised that teams be split at this stage to help increase the rate at which services are defined. A team that is decomposing a service should have

- The primary stakeholder or business owner of that area (identified in the service definition)
- A representative from all groups that interact with that service
- A facilitator
- Representatives from the business and technical community of that domain

Teams should then report back their results to the whole group. The timetable should be expected to be similar to that described in section 11.2.6, which means an event scheduled over 3 days at an enterprise level should only expect to reach level 2 at best.

If there are impacts on the Level 0 model these are dealt with immediately, the main facilitator and the main sponsor act as the veto on changes to the Level 0 model and it is important to ensure that teams do not "cry wolf". One successful way of doing this is to apply a similar system to that used in the National Football League[12] in that the teams have a number of challenges. A team in this case is given a number of challenges, normally 2 or 3, and if their challenge is accepted they keep that number, if a challenge is unsuccessful they lose one of their challenge rights, if a team loses all of its rights it **cannot** challenge the Level 0 diagram. Another option is to use a variation on the "sin-bin" approach from Rugby, Field Hockey and Ice Hockey, here on an unsuccessful challenge the team is excluded from making a challenge for a given period of minutes, normally between 5 and 10.

11.2.8 Understanding when the Diagram is done

A key part of facilitating the event is prevent the group going into details and trying to create complex process maps to connect the services. This is very much a secondary task, as is the question of "Where does my SAP system fit in this" or "We use .NET". The deliverables defined in section 11.2.6 are *all* that is required. The purpose of this event is to create a simple diagram that everyone can agree with. The more detail that is created the less clarity there will be in that diagram. The facilitator needs to drive towards this big picture and to judge the level of decomposition that is required. This level may be different depending on the part of the "tree" that it is in.

11.2.9 Publishing the big picture

Once the big picture has been created it needs to be published so everyone can see it. Make sure there is access to a large scale printer (A1 and potentially bigger) to print out the diagram on a single sheet. Multiple A3 sheets can do the job but its cleaner on one piece of paper. Computer monitors tend not to have either the size or resolution to display a full diagram in one image. The challenge at this stage is one of representation and information. We have used ovals to represent Services throughout this document as when conceptualising a framework they can present a more abstract notion than a more regimented regular shape. Ovals however are poor when it comes to representing nesting and the level of services that could exist at this level.

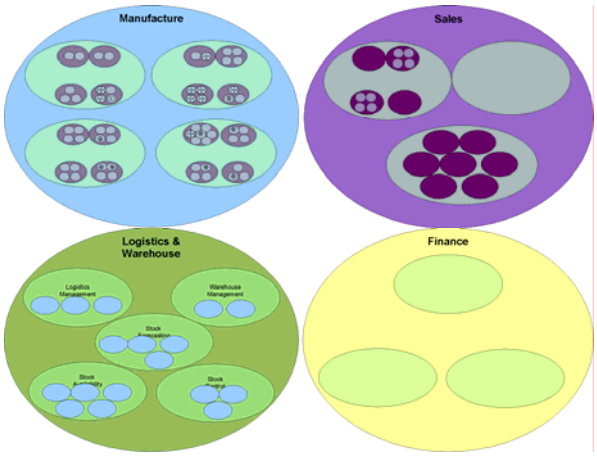

Figure 23 "Big Picture" in Ovals

Figure 23 shows the challenge. Simply put Ovals just don't tessellate. So putting text into the various boxes becomes harder and harder, but not impossible. If it is possible to represent the diagram in this way then that should be done, but for large enterprise and programme models it sometimes becomes necessary to move towards a simpler shape, the rectangle. This shift could also be done if there is a desire to represent the architecture not as something constant evolving and under review, but as a representation of what shall be. By using a more regimented shape it helps add formality to the same information.

Figure 24 has the same information but displayed in the different format. The key here is that people should be able to quickly walk up to a wall with the A1/A0 picture on and pin-point the area they need to consider. This is the picture that should be on every wall of every person that needs to know how things work together. Apart from the static images there should also be a website that guides people down. Standard organisational tools, such as the one in Microsoft Visio, can be used for this model to create a simple drill down and navigation mechanism.

Figure 24 "Big Picture" in Rectangles

Ideally there should be a tool that supports this approach, but as yet none in the market appear to be mature enough to provide the functionality required for this first diagram, let alone the refinements that come next. This first diagram is the big picture of the project, programme or enterprise, that gives context for all decisions. However the support services and identification of shared services has still to be completed.

11.2.10 End of the Event

The event has one further task beyond the creation of the diagram and the detail on the services. This is to define the timescales and next steps. These next steps can differ depending on the context and scale of the undertaking. For an enterprise model the key next element could be to identify the high level business processes that work between the services, for a project it might be to identify the current technology that is delivering some of the services. The critical element however is to keep the momentum high. Table 6 shows some potential next steps, this list is not exhaustive, some of which could be actioned during the final session.

Table 6 Potential Next Steps

Title	Description	Project or Enterprise
Business Process Map	Co-ordinating the various services into high level business process maps	Enterprise
Execution Process Map	Detailing how services are to be orchestrated via technology to deliver business processes	Project
Support Services	Identification of the support services (all categories) that flesh out the architecture	Both
Service Specification	Detailed specification of the contract for a set of services.	Project
Project Identification	Using the Service Map, allied to business strategy, to understand what the most appropriate first project will be.	Enterprise
Communication Plan	How to propagate the findings beyond the event group	Both
Change Process	Definition of how revisions will be made to the Service Architecture	Both
Visibility	Technical support to the Communication plan, and the collaborative and interactive tools to navigate the service architecture	Both
Service Classification	Classification of the services by various means, including business value, risk, potential to change.	Both

Enterprise Architecture	Creation of the Enterprise Architecture that will deliver the Service Architecture, this often covers the IT Strategy and technology roadmaps of the organisation	Enterprise
Solution Architecture	How a specific project will be delivered and the fully fleshed out architecture for its delivery	Project

It is important that after the session that the full findings are made available as quickly as possible. At most 2 working days after the end of the session, this is done to ensure the information is close at hand as soon as people start talking about what they did.

The collaborative information site that has been created to support the service architecture needs to be kept up to date. And as the support services are added they need to be added to the overall "big-picture" creating a new image that represents the "whole picture". The big picture still needs to be kept as it was as it represents the business view of the system, but it is critical that particularly on the technology side there is a view of all the potential services. Quite clearly this can be a huge picture on a complex project or enterprise.

At this stage the service architecture has first been exposed, it is now possible to attack some of the other tasks from the perspective of these services.

11.3 "Get" is not a Service

A common mistake when defining services is to have "services" called "GetCustomer" "UpdateCustomer" etc, these are **not** services, they are just invocation points "The why" elements on the services. The service is "Customer", which has an interaction point of "Get". Services are collections of processes, not individual processes themselves, hence the reason that "Get" is not a service. To help with this thinking consider about an old, manual organisation, they would use paper forms (with many copies) on the basis of which many different tasks were done. For example: a requisition was raised by writing the basic information on the form. The form was then sent to the order dept for processing, your manager and his assistant for filing, to finance for their records and to match up when invoices came in. These multiple copies of the forms are then acted upon in different ways by these departments. And so on…

This "hand over" between function is the ambition of a service architecture, while finance on this occasion did "receive requisition" the organisational function is finance, and its task was "receive requisition", the "what" is finance, the "who" is the supplier, the "why" is "match requisition to delivery", only when we consider the "how" do we worry about the actual physical handing over of the form.

12. Developing the complete architecture

A properly defined SOA is the start to any architecture work. What effectively the service model has defined is the context, concept and goals of the whole architecture. It has also described the various different domains and the different business drivers that they have. As the full architecture is created the focus is then, within a project.on understanding where these services "live" in the infrastructure of the business, on how they are to be actually delivered to users. Within an Enterprise Architecture is the process is understanding the business, IT and vendor strategies and determining the best technical and architectural standards and practices for the organisation to migrate to a full SOA. Architecture can also add all of the non-functional elements to the services,

their security constraint, performance requirements, redundancy and reliability and generally use the basic service architecture as the starting point for a fully fledged project or enterprise architecture. This is Service Architecture therefore represents the skeleton onto which the complete architecture, whether Enterprise or Solution, adds flesh, sinew and muscle.

It is quite normal that the definition of technical, and some support, services is left until the project architecture phase of the project, and it is certain that both enterprise and project architecture phases will modify some of the service understandings, in particular around shared services.

When attempting to move an organisation from "Big to Small"[13] the services architecture helps in two ways, firstly by identifying the small elements, and via the hierarchy helping to understand what the big picture is as well. The key when evolving the architecture is to pick a good practice guide and toolkit such as Capgemini's Integrated Architecture Framework, which for 10 years has had the concept of "service" at its heart.

13. Managing Change

With a properly implemented Service Architecture change becomes easier to manage .Because the services map to the business the change tends to be limited within those bounds, and where requests go across bounds it is a clear demonstration as to why it can be more complicated.

Another advantage of service architectures, when properly delivered, is that services can be maintained and released at different heartbeats, rather than having large maintenance cycles into which everything can fit. This again requires an increase in best practice over what many organisations do and requires robust automated testing and quality control.

14. Summary

This paper concentrates on the start of the service architecture, the most important element. And briefly explains how other elements can then benefit or be linked in from this approach. The key driver is to get the *services* right and ensure they represent what the business wants.

If you are serious about services architecture then it's important to create the services well before you think of technology. Remember what ***Service*** Oriented Architecture is about…

"It's the Services Stupid"

Table of Figures

Table of Tables

References

[1] IAF – Capgemini's Integrated Architecture Framework
[2] S. Jones "Toward an acceptable definition of service" IEEE Software May/June 2005 (Vol. 22 No. 3) pp 87-93
[3] P. Kroll and P. Krutchen "The Rational Unified Process Made Easy: A Practitioners Guide to the RUP", Addison Wesley, ISBN 0321166094
[4] K. Beck "Extreme Programming Explained – Embrace Change", Addison Wesley, ISBN 0321278658
[5] OASIS Blueprints group http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-blueprints and OASIS Reference Model group http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm
[6] J. A. Zachman, "A Framework for Information Systems Architecture," IBM Systems Journal 26, No. 3, 276-292 (1987).
[7] Martin Op 't Land – "Capgemini Usability of Capgemini's Integrated Architecture Framework (IAF) compared with the Extensible Architecture Framework (xAF)" – LAC 2004
[8] Petrovic, O - "Proceedings of the Twenty-Eighth Hawaii International Conference on System Sciences, 1995. Vol. IV."
[9] G. Geurts and A. Geelhoed "Business Process Decomposition and Service Identification Using Communication Patterns" – MSDN Journal 1 ,January 2004
[10] Bertrand Meyer: *Applying "Design by Contract*, in *Computer (IEEE)*, vol. 25, no. 10, October 1992, pages 40-51
[11] Proceedings. Thirteenth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises 14-16 June 2004
[12] NFL challenge and timeout rules - http://www.nfl.com/news/990526replaytechnology.html
[13] A. Mulholland – "Moving from Big to Small"